

\~15~

PAT-NO: JP411143511A

DOCUMENT-IDENTIFIER: JP 11143511 A

TITLE: NUMERICAL CONTROLLER

PUBN-DATE: May 28, 1999

INVENTOR-INFORMATION:

NAME	COUNTRY
HOSOKAWA, MASAHIKO	N/A
NAKAMURA, MINORU	N/A

ASSIGNEE-INFORMATION:

NAME	COUNTRY
FANUC LTD	N/A

APPL-NO: JP09316623

APPL-DATE: November 4, 1997

INT-CL (IPC): G05B019/18

ABSTRACT:

PROBLEM TO BE SOLVED: To provide a numerical controller different in command specification by which a command can be made in common and also the command to

be used therefor can be used independently of a hardware or software of the numerical controller.

SOLUTION: This numerical controller relieves a command and the difference of a numerical controller specification by an inherent command of a virtual numerical controller 10, makes the command in common and also forms a command of the numerical controller which does not depend on the hardware and software of the numerical controller by providing an application that includes a command which is used for the numerical controller and the controller 10 which is independent of the specification of a real numerical controller that is actually driven or executing the application that includes that includes the command which is used for the numerical controller and virtual numerical control which is independent of the specification of the real numerical controller that is practically driven.

COPYRIGHT: (C)1999,JPO

(11)特許出願公開番号

特開平11-143511

(43)公開日 平成11年(1999)5月28日

(51) Int.Cl.⁶
G 0 5 B 19/18

識別記号

F I
G 0 5 B 19/18

S

審査請求 有 請求項の数10 FD (全 20 頁)

(21)出願番号 特願平9-316623

(22)出願日 平成9年(1997)11月4日

(71)出願人 390008235

ファナック株式会社

山梨県南都留郡忍野村忍草字古馬場3580番地

(72)発明者 細川 匡彦

山梨県南都留郡忍野村忍草字古馬場3580番
地 ファナック株式会社内

(72)発明者 中村 稔

山梨県南都留郡忍野村忍草字古馬場3580番
地 ファナック株式会社内

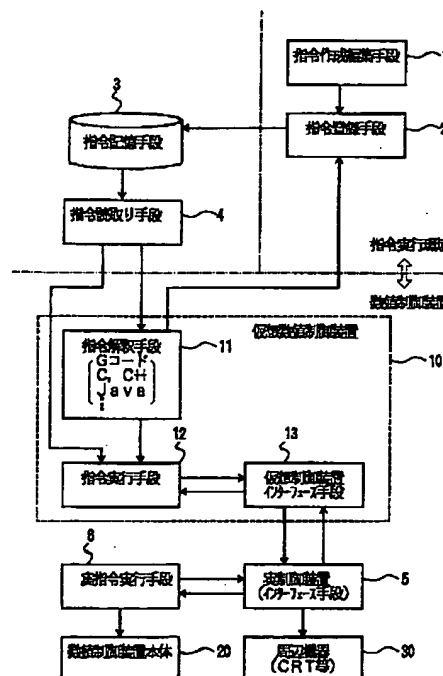
(74)代理人 弁理士 竹本 松司 (外4名)

(54)【発明の名称】 数値制御装置

(57) 【要約】

【課題】 指令の仕様の異なる数値制御装置において、指令を共通化することができ、又、数値制御装置に用いる指令を数値制御装置のハードウェアやソフトウェアに依存することなく用いることができる数値制御装置を提供する。

【解決手段】 数値制御装置において、数値制御装置に用いる指令を含むアプリケーション、及び実際に駆動する実数値制御装置の仕様と独立した仮想数値制御装置10を設けることによって、あるいは、数値制御装置に用いる指令を含むアプリケーション、及び実際に駆動する実数値制御装置の仕様と独立した仮想の数値制御を実行することによって、指令及び数値制御装置の仕様の相違を仮想数値制御装置の固有の指令によって緩和し、指令を共通化するとともに、数値制御装置のハードウェアやソフトウェアに依存しない数値制御装置の指令を形成する。



【特許請求の範囲】

【請求項1】 入力される指令と独立した固有の指令形態を有する仮想数値制御装置を備え、該仮想数値制御装置は入力した指令を固有の形態の指令に変換し、該変換した指令に基づいて実数値制御装置の実行を行う仮想数値制御装置を備える数値制御装置。

【請求項2】 実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置を備え、該仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、該指令の実行に基づいて実数値制御装置の実行を行う数値制御装置。

【請求項3】 入力される指令と独立し、又実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置を備え、仮想数値制御装置は入力した指令を固有の形態の指令に変換し、仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、該指令の実行に基づいて実数値制御装置の実行を行う数値制御装置。

【請求項4】 前記仮想数値制御装置は、任意の言語形式及び任意の動作仕様で作成された指令を解釈し、数値制御装置固有の形態の指令に変換する指令解釈手段を備える請求項1、または3記載の数値制御装置。

【請求項5】 前記仮想数値制御装置は、仮想数値制御装置に固有の形態に変換した指令を逐次読み込み実行する指令実行手段を備える請求項1、3又は4記載の数値制御装置。

【請求項6】 前記仮想数値制御装置は、仮想数値制御装置側のデータ仕様を実数値制御装置のデータ仕様に変換し、仮想数値制御装置と実数値制御装置との間のデータの整合性を調整する仮想数値制御装置インターフェース手段を備える請求項2、又は3記載の数値制御装置。

【請求項7】 前記任意の言語形式は、汎用プログラミング言語である請求項4記載の数値制御装置。

【請求項8】 仮想表示装置に対して表示指令を出力する仮想表示指令手段及び前記表示指令を実表示装置に対する表示指令に変換する表示指令変換手段と、仮想操作装置に対する操作指示を出力する仮想操作指示手段及び実操作装置から操作指示を取得し仮想操作装置に対する操作指示に変換する操作指示変換手段と、仮想通信装置を用いる手段及び仮想通信装置と実通信装置との間で通信データ交換を行う通信データ交換手段と、仮想数値制御装置と実数値制御装置との間でデータ交換を行う、仮想数値制御装置側のデータ交換手段及び実数値制御装置側のデータ交換手段の少なくとも1つを備えた数値制御装置。

【請求項9】 入力された指令を該指令と独立した仮想数値制御装置に固有の指令形態に変換し、該変換した指令を仮想数値制御装置上で実行することにより、実数値制御装置の実行を行う数値制御装置の実行方法。

【請求項10】 実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置上において仮想数値制

御装置に固有の形態の指令を実行し、該仮想数値制御装置上の指令実行に基づいて実数値制御装置の実行を行う数値制御装置の実行方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、数値制御装置に関し、指令の仕様にかかわらず動作可能な数値制御装置に関する。

【0002】

【従来の技術】一般に、数値制御装置では、加工指令及びそれに付随する補助指令、加工指令をカスタマイズするマクロ指令、表示と操作をカスタマイズするC言語指令等の各種の指令が用いられている。通常、この指令は、指令対象の数値制御装置に固有の仕様に従って作成されている。この指令の作成において、数値制御装置が異なると、同一の機能であっても指令方法が異なったり、指令方法そのものが定義されていない場合がある。従って、同一の指令を行わせる場合でも、指令対象の数値制御装置に応じて指令の仕様が異なる部分がある場合には、全く別に指令作成を行っている。

【0003】そのため、指令作成を行うには、指令対象の数値制御装置の仕様を十分に考慮する必要がある。

【0004】図9は従来の数値制御装置の指令の動きを説明するためのブロック概略図である。図9は、指令作成を主たる機能とする指令実行環境と、作成した指令により数値制御装置本体を実行して加工等の作業を行う数値制御装置とを含む構成を示している。指令実行環境は、指令作成編集手段1と指令登録手段2を備え、指令作成編集手段1で作成した指令を指令登録手段2を介して指令記憶手段3に記憶する。数値制御装置側には、指令記憶手段3、指令読取り手段4、指令解釈手段11、制御装置インターフェース手段15、指令実行手段16、数値制御装置本体20、周辺機器30を備える。

【0005】指令記憶手段3に格納された指令は、指令読取り手段4を介して指令実行手段16に送られる。この指令は数値制御装置本体20で実行し加工等の処理を行う。又、指令実行手段16はインターフェース手段となる制御装置15を介してCRTや入出力装置等の周辺機器30との間でデータ交換を行う。ここで、指令実行手段16による指令実行において、コンパイラあるいは指令解釈手段11でインタプリタで、数値制御装置本体20が実行可能な信号形態に変換している。

【0006】

【発明が解決しようとする課題】従来の数値制御装置では、上記したように、指令作成において、数値制御装置固有の仕様について多くの知識が必要であり、又、多大な時間と労力を要するという問題点がある。

【0007】数値制御装置を使用した工作機械では、機能の向上やコストの削減等の種々の理由から、別の数値制御装置を用いる場合がある。このような場合、指令の

仕様が数値制御装置によって異なるため、全く別の指令を作成する必要がある、この指令作成の工程は大きな負担となっている。

【0008】又、レトロフィット等により、加工装置等の駆動部分を生かし、数値制御装置を新しいものに交換する場合においても、指令の仕様が異なるため、従前の数値制御装置で長年蓄積された指令を交換後の数値制御装置の仕様に変更する必要がある。そのため、交換に多大な時間と労力を要する上に、使い慣れた仕様の指令が使え無くなるという問題点もある。

【0009】通常、数値制御装置の仕様は数値制御装置のメーカーが独自に行うため、数値制御装置を使用する側では、この数値制御装置の変更による負担を回避できないという問題がある。

【0010】図10は従来の数値制御装置における指令実行環境と数値制御装置との関係を説明するための概略図である。図10(a-1)、(a-2)、(a-3)は、指令実行環境aに対する数値制御装置A、B、Cの関係を示している。指令実行環境aは、数値制御装置Aに対応して指令作成を行うことによって、数値制御装置による加工を行うことができる。この指令実行環境aを数値制御装置Bあるいは数値制御装置Cに適用した場合には、指令の仕様が対応していないため、指令実行環境aを用いて数値制御装置Bあるいは数値制御装置Cによる加工を行うことはできない。

【0011】同様に、指令実行環境bは数値制御装置Bにのみ対応し、指令実行環境cは数値制御装置Cにのみ対応し、仕様が異なる数値制御装置での加工は行うことができない。

【0012】そこで、本発明は前記した従来の問題点を解決し、指令の仕様の異なる数値制御装置において、指令を共通化することができる数値制御装置を提供することを目的とし、又、数値制御装置に用いる指令を数値制御装置のハードウェアやソフトウェアに依存することなく用いることができる数値制御装置を提供することを目的とする。

【0013】

【課題を解決するための手段】本発明の数値制御装置は、数値制御装置において、数値制御装置に用いる指令を含むアプリケーション、及び実際に駆動する実数値制御装置の仕様と独立した仮想数値制御装置を設けることによって、あるいは、数値制御装置に用いる指令を含むアプリケーション、及び実際に駆動する実数値制御装置の仕様と独立した仮想の数値制御を実行することによって、指令及び数値制御装置の仕様の相違を仮想数値制御装置の固有の指令によって緩和し、指令を共通化するとともに、数値制御装置のハードウェアやソフトウェアに依存しない数値制御装置の指令を形成する。

【0014】これによって、従来行っていた数値制御装置毎に指令を作り直す工程やそれらに係わる保守の工程

を低減することができる。又、実装される数値制御装置の種類や選択可能な機械メカと、その機械を使用する機械加工工場等のエンドユーザで作成されるアプリケーションプログラムの互換性、及び保守工程の低減を保証することができる。

【0015】又、仮想数値制御装置及び仮想数値制御装置による指令の実行を行うことによって、従来の数値制御装置で行っていた数値制御装置側の制御ソフトウェアと使用者側のアプリケーションプログラムとの結合工程を省くことができる。

【0016】そこで、本発明の数値制御装置は、入力される指令と独立した固有の指令形態を有する仮想数値制御装置を備え、この仮想数値制御装置は入力した指令を固有の形態の指令に変換し、変換した指令に基づいて実数値制御装置の実行を行う仮想数値制御装置を備える数値制御装置である。これによって、仮想数値制御装置を指令を含むアプリケーションと独立したものとし、入力した指令を仮想数値制御装置に固有の形態の指令に変換し、この変換した指令に基づいて実数値制御装置の実行を行う。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、数値制御装置の実行を行うことができる(請求項1に対応)。

【0017】又、本発明の数値制御装置は、実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置を備え、この仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、この実行に基づいて実数値制御装置とデータ交換を行い、実数値制御装置の実行を行う数値制御装置である。これによって、数値制御装置のハードウェアやソフトウェアに依存することなく指令に実行を行うことができる(請求項2に対応)。

【0018】又、本発明の数値制御装置は、入力される指令と独立し、又実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置を備え、仮想数値制御装置は入力した指令を固有の形態の指令に変換し、仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、この指令の実行に基づいて実数値制御装置とデータ交換を行い、実数値制御装置の実行を行う数値制御装置である。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、数値制御装置の実行を行うことができるとともに、数値制御装置のハードウェアやソフトウェアに依存することなく指令に実行を行うことができる(請求項3に対応)。

【0019】又、本発明の数値制御装置において、仮想数値制御装置は、任意の言語形式及び任意の動作仕様で作成された指令を解釈し、数値制御装置固有の形態の指令に変換する指令解釈手段を備える構成とする。指令解釈手段は、入力される指令を解釈して数値制御装置固有の形態の指令に変換する機能及び両指令間の対応関係データを備え、入力される指令のプログラミング言語や仕

様に対応して解釈を行い、数値制御装置固有の形態の指令に変換する。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、仮想数値制御装置の実行を行うことができる。

【0020】なお、指令を解釈する機能及び数値制御装置固有の形態の指令へ変換する機能は、ソフトウェアにより行うことができ、又、両指令間の対応関係データは、ROM等の記憶装置に格納することができる（請求項4に対応）。

【0021】又、本発明の数値制御装置において、仮想数値制御装置は、仮想数値制御装置に固有の形態に変換した指令を逐次読み込み実行する指令実行手段を備える構成とする。指令実行手段は、前記指令解釈手段で解釈し変換した指令を、通常のプロセッサの動作と同様に逐次読み出して実行する。このとき、指令実行手段は仮想数値制御装置に固有の指令を実行するため、指令の仕様や指令を記述するプログラミング言語に依存せずに行うことができる（請求項5に対応）。

【0022】又、本発明の数値制御装置において、仮想数値制御装置は、仮想数値制御装置側のデータ仕様を実数値制御装置のデータ仕様に変換し、仮想数値制御装置と実数値制御装置との間のデータの整合性を調整する仮想数値制御装置インターフェース手段を備える構成とする。仮想数値制御装置インターフェース手段は、前記指令実行手段の指令の形態を実数値制御装置が実行可能な形態に変換し、変換した指令を実数値制御装置に渡す。仮想数値制御装置インターフェース手段は、仮想数値制御装置に固有の指令形態と実数値制御装置側の指令形態との対応関係を有しており、両者間の整合性をとる。これによって、数値制御装置のハードウェアやソフトウェアに依存することなく指令に実行を行うことができる（請求項6に対応）。

【0023】又、本発明の数値制御装置において、入力される指令を表すプログラミング言語形式は、汎用プログラミング言語とするものである。汎用プログラミング言語の例としては、例えばC++言語やJAVA言語を用いることができる。この他、入力される指令をGコード等のコード命令とすることもできる（請求項7に対応）。

【0024】さらに、本発明の数値制御装置において、仮想表示装置に対して表示指令を出力する仮想表示指令手段及び前記表示指令を実表示装置に対する表示指令に変換する表示指令変換手段と、仮想操作装置に対する操作指示を出力する仮想操作指示手段及び実操作装置から操作指示を取得し仮想操作装置に対する操作指示に変換する操作指示変換手段と、仮想通信装置を用いる手段及び仮想通信装置と実通信装置との間で通信データ交換を行う通信データ交換手段と、仮想数値制御装置と実数値制御装置との間でデータ交換を行う、仮想数値制御装置側のデータ交換手段及び実数値制御装置側のデータ交換

手段の少なくとも1つを備えた構成とすることができる。

【0025】この構成によれば、数値制御装置の表示機能や操作機能や通信機能や、その他数値制御装置に係わるデータ交換の機能を、仮想数値制御装置上で実行して実数値制御装置側に反映させることができる（請求項8に対応）。

【0026】又、本発明では、上記仮想数値制御装置の機能を実現させるためのプログラムをプロセッサ読み取り可能な記憶媒体に記憶することができる。この記憶媒体の一の形態は、数値制御装置が備えるプロセッサに、入力された指令を該指令と独立した仮想数値制御装置に固有の指令形態に変換する機能、前記変換した指令を仮想数値制御装置上で実行する機能、前記実行に基づき実数値制御装置の実行を行わせる機能を実現させるためのプログラムを記憶したプロセッサ読み取り可能な記憶媒体である。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、数値制御装置の実行を行うことができる機能をプロセッサに実行させることができる。又、記憶媒体の一の形態は、数値制御装置が備えるプロセッサに、実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置上において仮想数値制御装置に固有の形態の指令を実行させる機能、前記仮想数値制御装置上の指令実行に基づいて実数値制御装置の実行を行わせる機能を実現させるためのプログラムを記憶したプロセッサ読み取り可能な記憶媒体である。これによって、数値制御装置のハードウェアやソフトウェアに依存することなく指令に実行を行うことができる機能をプロセッサに実行させることができる。

【0027】又、本発明は、入力された指令を該指令と独立した仮想数値制御装置に固有の指令形態に変換し、該変換した指令を仮想数値制御装置上で実行することにより、実数値制御装置の実行を行う数値制御装置の実行方法である。この方法によって、仮想数値制御装置を指令を含むアプリケーションと独立したものとし、入力した指令を仮想数値制御装置に固有の形態の指令に変換し、この変換した指令に基づいて実数値制御装置の実行することができる。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、数値制御装置の実行を行うことができる（請求項9に対応）。

【0028】又、本発明は、実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置上において仮想数値制御装置に固有の形態の指令を実行し、該仮想数値制御装置上の指令実行に基づいて実数値制御装置の実行を行う数値制御装置の実行方法である。この方法によって、数値制御装置のハードウェアやソフトウェアに依存することなく指令に実行を行うことができる（請求項10に対応）。

【0029】従って、本発明によれば、アプリケーションの共通性や保守性、又異なる数値制御装置との間での

互換性を保証することができる。又、数値制御装置の開発工数や保守工数を軽減することができる。

【0030】

【発明の実施の形態】以下、本発明の実施の形態を図を参照しながら詳細に説明する。本発明の実施の形態の構成例について、図1の本発明の数値制御装置を説明するための概略ブロック図を用いて説明する。図1において、指令作成を主たる機能とする指令実行環境と、作成した指令により数値制御装置本体を実行して加工等の作業を行う数値制御装置とを含む構成を示している。指令実行環境は、指令作成編集手段1と指令登録手段2を備え、指令作成編集手段1で作成した指令を指令登録手段2を介して指令記憶手段3に記憶する。数値制御装置側には、指令記憶手段3、指令読取り手段4、仮想数値制御装置10、実制御装置インターフェース手段5、実指令実行手段6、数値制御装置本体20、周辺機器30を備え、仮想数値制御装置10は指令解釈手段11、指令実行手段12、及び仮想制御装置インターフェース手段13の各機能を備える。なお、指令記憶手段3、指令読取り手段4を指令実行環境側とする構成、又は、指令作成編集手段1、指令登録手段2を数値制御装置側とする構成とすることもできる。

【0031】指令記憶手段3に格納された指令は、指令読取り手段4を介して仮想数値制御装置10に送られる。この指令が加工指令の場合は、数値制御装置本体20で実行可能な指令に変換されて、数値制御装置本体20で処理される。実指令実行手段6は実制御装置インターフェース手段5を介してCRTや入出力装置等の周辺機器30との間でデータ交換を行う。

【0032】仮想数値制御装置10は、指令実行環境と実行手段と実際の実数値制御装置との仕様の差異を吸収する目的で設けられる構成及び機能である。この、仮想数値制御装置10の部分は、数値制御装置毎に、数値制御装置の仕様や用いられる可能性のある指令の仕様やプログラミング言語に対応して開発し実装することができる。また、仮想数値制御装置10の機能をプロセッサに実行するに用いるプログラムをプロセッサ読み取り可能な記憶媒体に記憶させ、数値制御装置や指令の仕様を拡張あるいは変更した場合に、記憶媒体により付加あるいは変更することができる。又、ある数値制御装置で開発した仮想数値制御装置の機能を、他の数値制御装置に転用することもできる。

【0033】指令解釈手段11は、入力された指令について文法解釈を行い、指令を仮想数値制御装置に固有の形態に変換し、これによって入力される指令に依存しない実行を可能なものとする。例えば、入力される指令の表示形態について、Gコード、C言語、C++言語、JAVA言語等のプログラミング言語を想定する場合には、指令解釈手段11にこれらのプログラミング言語の文法解釈を行う機能を持たせ、又、動作仕様についても

想定される仕様に対応した解釈機能を持たせる。

【0034】指令解釈手段11は、指令読取り手段4から読取った指令を前記機能によって解釈し、仮想数値制御装置に固有の形態の指令に変換し、指令実行手段12に渡す。なお、指令解釈手段11で解釈した指令を指令登録手段2を介して指令記憶手段3に格納し、あらかじめ実行する指令について解釈処理を完了させておき、解釈後の指令をまとめて指令実行手段12で実行する構成とすることもできる。この構成によれば、実行の高速化を図ることができる。

【0035】指令実行手段12は、仮想数値制御装置10のプロセッサ部分であり、指令解釈手段11で処理された指令を、通常のマイクロプロセッサと同様にして逐次読み込んで実行する。この指令実行手段12で実行される指令は、仮想数値制御装置10に共通に用いられる仕様である。従って、仮想制御装置インターフェース手段13は、指令実行手段12からの指令をそのまま指令どおりに実行することができる。

【0036】仮想制御装置インターフェース手段13は、実制御装置インターフェース手段6と共に、仮想数値制御装置10と実数値制御装置との間のデータの交換を行うインターフェース部分である。なお、ここでは、実数値制御装置は、実制御指令インターフェース手段5、実指令実行インターフェース手段6、数値制御装置本体20、及び周辺機器30を含めて部分を示している。

【0037】仮想制御装置インターフェース手段13は、指令実行手段12から入力された指令に対して、実指令実行インターフェース手段6の仕様に変換して出力する。又、実指令実行インターフェース手段6から入力されたデータを仮想数値制御装置の仕様に変換し、指令実行手段12に出力する。仮想制御装置インターフェース手段13は、上記機能を達成するために、仮想数値制御装置と実数値制御装置との仕様の対応関係に関するデータを備え、このデータに基づいて変換を行う。従って、この仮想制御装置インターフェース手段13は、実数値制御装置との間のデータの入出力の整合性をとる役目をしている。

【0038】仮想数値制御装置10において、指令解釈手段11は指令実行環境側の仕様と仮想数値制御装置側の仕様との対応関係、及び変換機能を備え、仮想制御装置インターフェース手段13は、仮想数値制御装置側の仕様と実数値制御装置側の仕様指令との対応関係、及び変換機能を備えており、これらの対応関係、及び変換機能はプロセッサで実行するのに用いるプログラムとすることができ、プロセッサ読み取り可能な記憶媒体に記憶させることができる。従って、これらの対応関係、及び変換機能の全部あるいは一部を記憶媒体の形態で移動させ、他の数値制御装置に移植することもできる。

【0039】なお、実指令実行インターフェース手段6

は、仮想制御装置インターフェース手段13からの指令を実数値制御装置の仕様に交換するインターフェース部分である。

【0040】図1に示す構成の数値制御装置中の仮想数値制御装置の動作について、図2のフローチャートを用いて説明する。仮想数値制御装置10は、指令読取り手段4から入力される指令について、用いられているプログラミング言語や仕様等の態様を認識し、指令解釈手段11で行う解釈処理に対応する(ステップS1)。指令読取り手段4から指令が読み取られて指令が入力されると(ステップS2)、指令解釈手段11はこの指令について文法解釈等の解読処理を行い、仮想数値制御装置に固有の機械語(ニューモニックコード)に変換する。変換した機械語は、指令実行手段12に引き渡す。なお、指令解釈手段11で変換した機械語を、指令登録手段2を介して指令記憶手段3に蓄積しておき、指令実行手段12に渡す処理を行うこともできる。この処理によれば、処理の高速化を図ることができる(ステップS4)。

【0041】処理実行手段12は、入力した機械語を逐次実行し、その実行を仮想制御装置インターフェース手段13に渡す(ステップS5)。仮想制御装置インターフェース13は、渡された実行を実数値制御装置で実行することができる仕様に交換して実制御装置インターフェース手段5に渡し、実数値制御装置での実行を行わせる(ステップS6)。

【0042】本発明の数値制御装置の構成及び制御方法によれば、処理実行環境側と実際の数値制御装置側の仕様が相違する場合においても、それらの相違に対応して指令を作成することなく実行することができる。図3は本発明の数値制御装置による指令実行環境と数値制御装置との関係を説明するための図である。

【0043】図3(a-1)、(a-2)、(a-3)において、同一の指令実行環境aを異なる数値制御装置A、B、Cに適用させることができる。同様に、同一の指令実行環境b、cを異なる数値制御装置A、B、Cに適用させることもできる。又、同一の数値制御装置Aに対して異なる指令実行環境a、b、cを適用させることができ、同様に、同一の数値制御装置B、Cに対して異なる指令実行環境a、b、cを適用させることができる。

【0044】図4は、仮想数値制御装置側と実際の数値制御装置(実数値制御装置で表す)との間の仕様の相違を吸収するための構成を説明するための概略図であり、*

*図1で示す仮想制御装置インターフェース手段及び実制御装置インターフェース手段が備える一構成例を示している。

【0045】図4において、仮想数値制御装置側は仮想制御装置側データ交換手段を備え、これによって仮想数値制御装置側のデータを実数値制御装置側に対応した仕様に交換する。一方、実数値制御装置側は実制御装置側データ交換手段を備え、これによって実数値制御装置側のデータを仮想数値制御装置側に対応した仕様に交換する。これによって、仮想数値制御装置側と実数値制御装置側でデータ仕様が異なる場合であっても、データの相互交換を行うことができる。

【0046】例えば、表示機能については、仮想数値制御装置側の仮想表示装置に対して表示指令を行う仮想表示指令手段と、仮想表示手段に対する表示指令を実際の表示装置(実表示装置)に対する指令に変換する表示指令変換手段を仮想数値制御装置側に設け、これによって実表示装置の実行を可能とする。又、操作機能については、仮想数値制御装置側の仮想操作装置に対して操作指令を行う仮想操作指令手段と、仮想操作手段に対する操作指令を実際の操作装置(実操作装置)に対する指令に変換する表示操作変換手段を仮想数値制御装置側に設け、これによって実操作装置の実行を可能とする。さらに、通信機能については、仮想数値制御装置側の仮想通信装置を利用する仮想通信利用手段に対してデータ交換を行う通信データ交換手段を仮想数値制御装置側に設け、これによって実通信装置の実行を可能とする。

【0047】なお、指令解釈手段11はソフトウェアによる文法解釈処理の他に、Gコード等のコード指令の場合には対応コード表を用いた構成とすることができる。図5は、対応コード表を用いた構成である。図5において、指令解釈手段11は、Gコードと仮想数値制御装置に固有の機械語(ニューモニックコード)との対応データを記憶部11bに備え、読み出し部11aにより読み出す構成とすることができる。

【0048】次に、指令を汎用プログラミング言語で記述した場合の動作例について説明する。以下、汎用プログラミング言語として、C++言語を用いた例とJAV A言語を用いた例について説明する。

【0049】なお、以下の例では、指令として位置表示指令の例について示す。

【0050】以下に示すプログラム例は、C++言語を用いた例である。

```
001: /*-----
002: * 仮想数値制御装置によるCNCの絶対位置の表示の例
003: *
004: *
005: *-----*/
006: #include "CncObject.h"
```



```

11
007: void main (void) {
008:     /*CNCオブジェクトのインスタンス生成*/
009:     CncObject cnc ;
010:
011:     /*絶対位置の取得*/
012:     cnc.GetAbsPos ( (short) -1) ;
013:
014:     /*制御軸数分処理を繰り返し、軸名称とその軸の絶対位置を
        表示する*/
015:     for ( int i=0;i<cnc.AxisNum : i++) {
016:         cnc.out.println (cnc.AxisName [i] +""+cnc.AbsPos [i] ) :
017:     }
018: }

```

12

上記プログラム例において、001行から005行は指令に影響しないコメント行である。006行はCncObjectの定義ファイルの読み込みを指定している。007行は関数の始まりであり、main関数はプログラムを開始する関数である。

【0051】009行はクラスCncObjectの実体を定義する部分である。ここで、CncObjectは、数値制御装置を仮想化したオブジェクトであり、図1に示す仮想制御装置インターフェース手段を呼び出すメソッドを持つ。実際の数値制御装置の通信手段とて通信されるデータ構造は、CncObjectが管理することによって、指令から見るとどのような数値制御装置でも同じ種類の制御装置として扱うことができる。

【0052】012行では、CncObjectの絶対位置取得メソッドであるGetAbsPos()を実行している。このメソッドを実行することでCncObject内の絶対位置のデータの更新が行われる。

【0053】015行では、制御軸数分だけ繰り返し実行するためのfor文である。cnc.AxisNumは009行のCncObjectの実体定義時に設定される制御軸数変数である。制御軸数や軸名称のような静的なデータは実体定義時に自動的に設定される。

【0054】016行は表示を行う関数である。メソッド名Cnc.out.println()は表示手段である。数値制御装置とのインターフェースにより実際の表示装置への指令に変換される。cnc.AxisName[]は数値制御装置の軸名称の配列変数であり、015行のように009行での実体定義時に設定される。cnc.AbsPos[]は数値制御装置の絶対位置の配列変数である。012行のように、メソッドGetAbsPos()を実行することで、その内容が更新され*

する。

【0055】上記プログラム例の指令の動作を、図1、図6、図7を参照して説明する。指令は指令登録手段2を通じて不揮発性メモリやハードディスク等の指令記憶手段3に格納される。指令解釈手段11は、指令読取り手段4を通じて指令記憶手段3中の指令を読み出す。指令解釈手段11が読み出した指令は通常のテキスト形式であるため、規定された文法に従って指令を解釈する。この指令解釈手段4により、指令は指令実行手段12への命令に翻訳される。この命令は通常のマイクロプロセッサと同様に機械語（ニューモニックコード）である。指令実行手段12は逐次機械語を実行する。入出力指令の機械語がある場合には、接続されている仮想制御装置インターフェース手段13に必要な引数を渡して処理を委託する。

【0056】この処理において、機械語となることでソフトウェアでもハードウェアでも処理を行うことができ、数値制御装置の性能やコスト等の条件に応じてソフトウェアとハードウェアの何れの形式で実装するかを選択することができる。

【0057】仮想制御装置ソフトウェアインターフェース手段13は、数値制御装置が入出力可能とする全データについて変換可能である。なお、以下では、前記した制御軸数、制御軸名称、及び絶対位置の各データの取得機能について説明する。

【0058】仮想制御装置インターフェース手段13は、指令上はCncObjectとして指令される。ここで、CncObjectの仕様は以下の表の通りである。

【0059】

【表1】

概要	数値制御装置と情報交換を行うクラス
メソッド	GetAbspos(軸番号) 数値制御装置の絶対位置を取得する。軸番号を引数とし、指定された軸の絶対位置をAbsPos[]配列変数を格納する。軸番号に-1が指定された場合には、全軸の絶対位置を取得する。
変数	AbsPos[] 数値制御装置の絶対位置が格納される配列変数 AxisName[] 数値制御装置の軸名称が格納される配列変数 AxisNo 数値制御装置の制御軸数が格納される配列変数

図6において、仮想制御装置インターフェース手段13において、009行のようにCncObjectのクラスの実体の定義が行われると、自動的に数値制御装置内の制御軸数データと軸名称データとを取得し、それぞれAxisNo、AxisName[]変数に格納される。実際には、この他にもハードウェアの実装状態等の動的に変化しないデータを取得する。

【0060】図7において、CncObjectのメソッドが指令されると、対応した仮想制御装置インターフェース手段13の関数が実行される。メソッドGetAbsPos()が指令されると、仮想制御装置インターフェース手段13に対応する絶対位置取得関数が実行される。絶対位置取得関数は、実際の装置の実指令実行手段に対応した指令に変換し、実指令実行手段6に指令を行う。実指令実行手段6は指令を実行し、実制御装置インターフェース手段5と通信を行い、絶対位置データを取得する。取得された絶対位置データは、逆の順序で仮想制御装置インターフェース手段13まで戻され、AbsPos[]配列変数に格納される。

*【0061】ここで、実制御装置インターフェース手段5は、数値制御装置固有の指令仕様を備える。仮想制御装置インターフェース手段13は、この数値制御装置による仕様の差異を吸収する。

20 【0062】図8は、数値制御装置側と使用者側の分担を説明するための図である。使用者側が用意するものはクラスMAINのみであり、このクラスMAINの部分は数値制御装置の種類にかかわらず実行可能である。数値制御装置側は、使用者側とのインターフェースであるクラスCncObjectの仕様を満たすように仮想制御装置インターフェース手段の部分を実装する。これによって、数値制御装置が異なる場合でも、クラスCncObjectで定義されたメソッドを十分に活用し、アプリケーションを構成することができる。このアプリケーションのプログラムは、数値制御装置に依存せず、又、再コンパイル等の処理を要することなく実行することができる。

【0063】以下に示すプログラム例は、J A V A言語を用いた例である。

```

001: /*-----
002: * J A V AによるCNCの絶対位置の表示の例
003: *
004: *
005: *-----*/
006: class Main {
007:     public static void main (String [] args) {
008:         try {
009:             /*CNCオブジェクトのインスタンス生成*/
010:             CncObject cnc=new CncObject ();
011:
012:             /*絶対位置の取得*/
013:             ret=cnc.GetAbsPos ( (short) -1) ;
014:
015:             /*制御軸数分処理を繰り返し、軸名称とその軸の絶対位置を
表示する*/

```

```

15
016:      for ( int i=0;i<cnc.AxisNum : i++) {
017:          System.out.println (cnc.AxisName [i] +" "+cnc.AbsPos [i]);
018:      }
019:  }
020:      /*エラーハンドラ*/
021:      catch (Exception e) {
022:          System.out.println (" In Java : %n"+e);
          /*エラーメッセージ表示*/
023:      }
024:  }
025:}

```

上記プログラム例において、001行から005行は指令に影響しないコメント行である。006行はクラス宣言の始まりであり、ここではクラス名をMainとしている。007行はクラスのメソッドの定義の始まりであり、ここではメソッド名をmainとしている。

【0064】008行はエラー処理の範囲を決めるために記述される部分である。008行から019行の範囲でエラーが発生した場合、021行から023行のエラーハンドラ処理に制御が移る。

【0065】010行はクラス名CncObjectの実体を定義する部分である。ここで、CncObjectは、数値制御装置を仮想化したオブジェクトであり、JAVAと数値制御装置のインターフェースとなる。実際の数値制御装置の通信手段とで通信されるデータ構造は、CncObjectが管理することによって、JAVAから見るとどのような数値制御装置でも同じ種類の制御装置として扱うことができる。

【0066】013行では、CncObjectの絶対位置取得メソッドであるGetAbsPos()を実行している。このメソッドを実行することでCncObject内の絶対位置のデータの更新が行われる。

【0067】016行では、制御軸数だけ繰り返し実行するためのfor文である。cnc.AxisNumは010行のCncObjectの実体定義時に設定さ*

*れる制御軸数変数である。制御軸数や軸名称のような静的なデータは実体定義時に自動的に設定される。

【0068】017行は表示を行う関数である。メソッド名System.out.println()はJAVAで定義される標準の表示手段である。JAVAは仮想的な表示装置を前提としているため、数値制御装置とのインターフェースにより実際の表示装置への指令に変換される。cnc.AxisName[]は数値制御装置の軸名称の配列変数であり、016行のように010行での実体定義時に設定される。cnc.AbsPos[]は数値制御装置の絶対位置の配列変数である。013行のように、メソッドcnc.GetAbsPos()を実行することで、その内容が更新される。

【0069】021行から023行はエラーハンドラ処理である。022行にエラー内容を表示するための手段を記述している。

【0070】次に、CncObjectについて説明する。以下では、説明の都合上前述のメソッドと変数のみ説明するが、実際の装置では、数値制御装置の入出力可能な全データについて交換可能である。

【0071】クラスCncObjectの仕様を以下の表2に示す。

【0072】

【表2】

概要	数値制御装置と情報交換を行うクラス
メソッド	GetCncInfo() 数値制御装置の制御軸数及び軸名称を取得し、それぞれAxisNo変数及びAxisName[]配列変数に格納する。 GetAbsPos(軸番号) 数値制御装置の絶対位置を取得する。軸番号を引数とし、指定された軸の絶対位置をAbsPos[]配列変数を格納する。軸番号に-1が指定された場合には、全軸の絶対位置を取得する。
変数	AbsPos[] 数値制御装置の絶対位置が格納される配列変数 AxisName[] 数値制御装置の軸名称が格納される配列変数 AxisNo 数値制御装置の制御軸数が格納される配列変数

この仕様は数値制御装置に依存しない。

* ectの例である。

【0073】以下のプログラムは、クラスCncObj *

```

001: /*-----
002: *class CncObject
003: *
004: *
005: *-----*/
006: class CncObject {
007:     public native short GetCncInfo();
008:     public native short GetAbsPos(short axis);
009:     public final static int MAX_AXIS=24;
010:     public long AbsPos[] = new long [ MAX_AXIS];
011:     public str AxisName[] = new str [ MAX_AXIS];
012:     public long AxisNo;
013:
014:     public static void CncObject() {
015:         GetCncInfo();
016:     }
017:
018:     static {
019:         System.loadLibrary(" MyImpOfCncObject");
020:     }
021: }

```

上記プログラム例において、001行から005行はコメント行である。006行はCncObjectの宣言開始を表している。007行はメソッドの定義であり、nativeという修飾子によって、メソッドの実体はJ A V AではなくJ A V Aの実行環境が実装される環境である数値制御装置の関数となる。これらがJ A V A実行環境と数値制御装置とのインターフェースとなり、J A V Aでの仮想的な数値制御装置と実際の数値制御装置との橋渡しを行う。ここでGetCncInfo()は、数値制御装置内の制御軸数データと軸名称データと※50

※を取得し、それぞれAxisNo、AxisName[]変数に格納する。

【0074】008行は007行と同様に数値制御装置の関数である。GetAbsPos()は数値制御装置内の絶対位置データを獲得し、AbsPos[]変数に格納する。

【0075】009行は、最大制御軸数を表す定数であり、このクラスで扱える最大制御軸数を表している。010行は、絶対位置を表す配列変数であり、008行で定義されるメソッドGetAbsPos()により更新

される。

【0076】011行は軸名称を表す配列変数、012行は制御軸数を表す変数である。これらは実体定義時に実行されるメソッドCncObject()により実行されるメソッドGetCncInfo()で設定される。

【0077】014行から016行はJAVAの実行環境から数値制御装置の関数を実行するためのライブラリを実行することを宣言するメソッドである。この宣言でのライブラリが数値制御装置の関数をJAVAから実行

【0078】図8において、使用者側が用意するものはクラスMAINのみであり、このクラスMAINの部分は数値制御装置の種類にかかわらず実行可能である。数値制御装置側は、使用者側とのインターフェースであるクラスCncObjectの仕様を満たすように仮想制御装置インターフェース手段の部分を実装する。数値制御装置が異なる場合でも、クラスCncObjectあるいはライブラリMyImpOfCncObjectでインターフェースを同一にすればよい。

【0079】これによって、数値制御装置が異なる場合でも、クラスCncObjectで定義されたメソッドを十分に活用し、アプリケーションを構成することができる。このアプリケーションのプログラムは、数値制御装置に依存せず、又、再コンパイル等の処理を要することなく実行することができる。

【0080】

【発明の効果】上記で説明したように、本発明の数値制御装置によれば、指令の仕様の異なる数値制御装置において、指令を共通化することができ、又、数値制御装置に用いる指令を数値制御装置のハードウェアやソフトウェアに依存することなく用いることができる。

【図面の簡単な説明】

【図1】本発明の数値制御装置を説明するための概略ブロック図である。

【図2】本発明の数値制御装置中の仮想数値制御装置の

動作を説明するためのフローチャートである。

【図3】本発明の数値制御装置による指令実行環境と数値制御装置との関係を説明するための図である。

【図4】本発明の仮想数値制御装置側と実際の数値制御装置との間の仕様の相違を吸収するための構成を説明するための概略図である。

【図5】本発明の指令解釈手段の対応コード表を用い構成を説明するための図である。

【図6】プログラム例の指令の動作を説明するための図である。

【図7】プログラム例の指令の動作を説明するための図である。

【図8】数値制御装置側と使用者側の分担を説明するための図である。

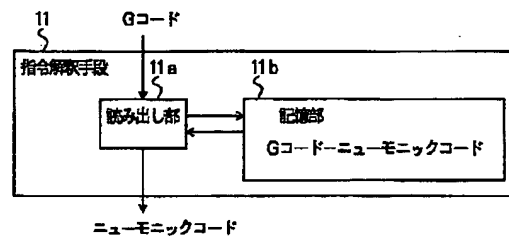
【図9】従来の数値制御装置の指令の動きを説明するためのブロック概略図である。

【図10】従来の数値制御装置における指令実行環境と数値制御装置との関係を説明するための概略図である。

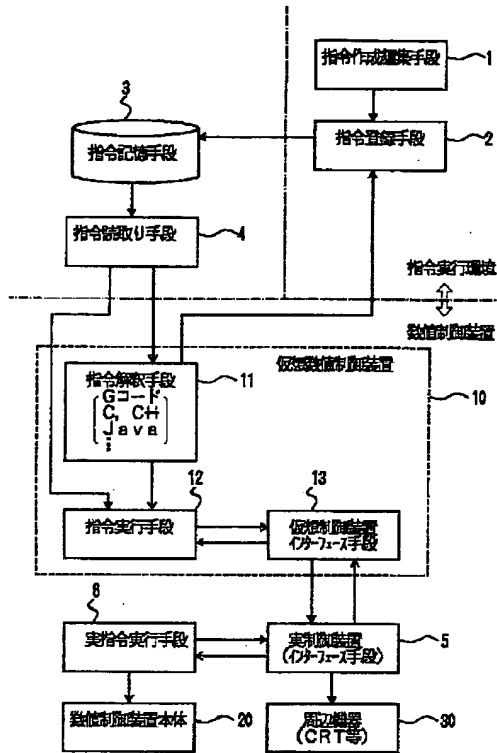
【符号の説明】

- 1 指令作成編集手段
- 2 指令登録手段
- 3 指令記憶手段
- 4 指令読取り手段
- 5 実制御装置インターフェース手段
- 6 実指令実行手段
- 10 仮想数値制御装置
- 11 指令解釈手段
- 11a 読み出し部
- 11b 記憶部
- 12 指令実行手段
- 13 仮想制御装置インターフェース手段
- 15 制御装置インターフェース手段
- 16 指令実行手段
- 20 数値制御装置本体
- 30 周辺機器

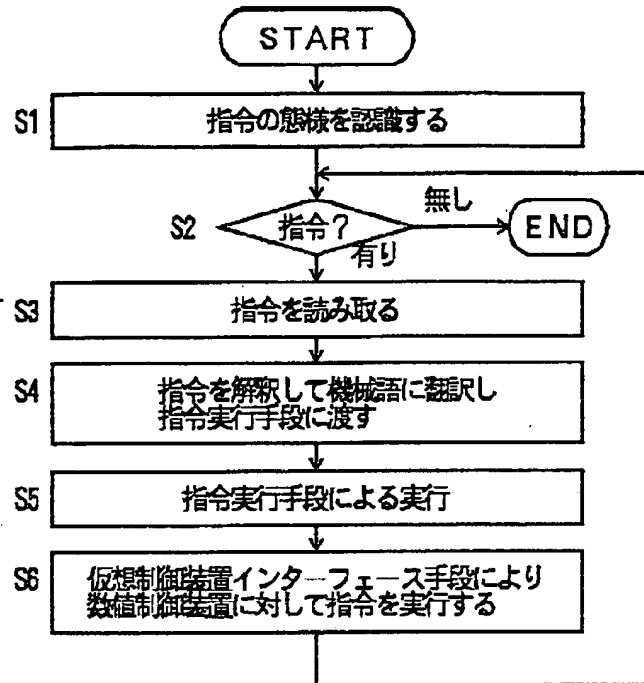
【図5】



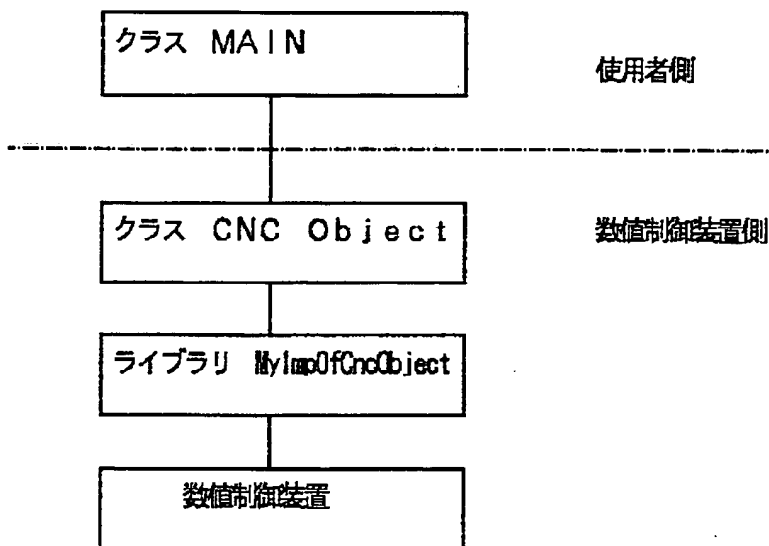
【図1】



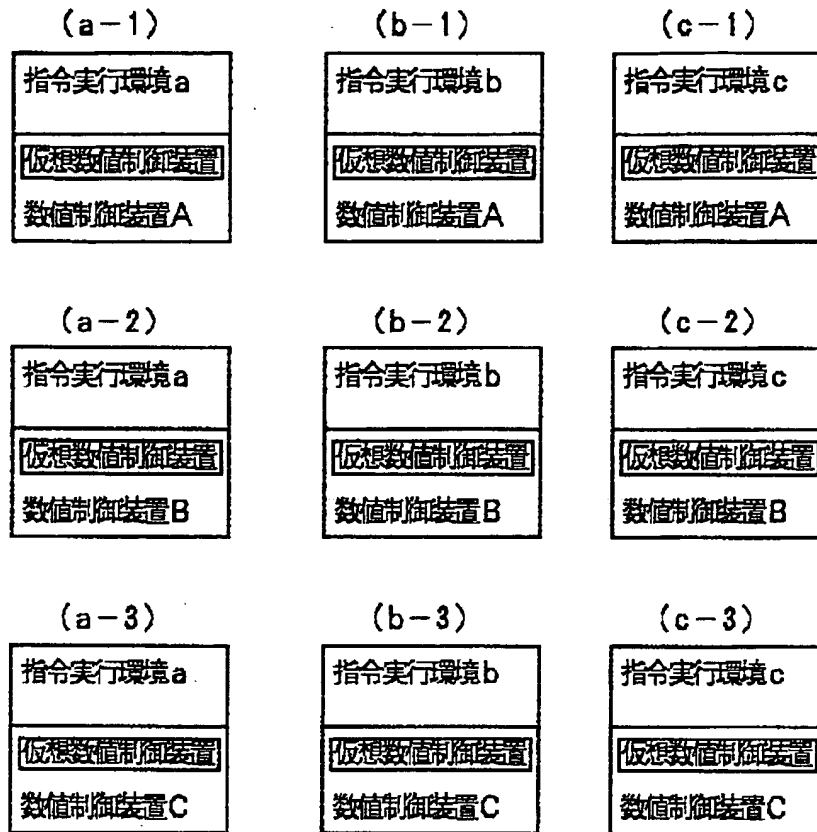
【図2】



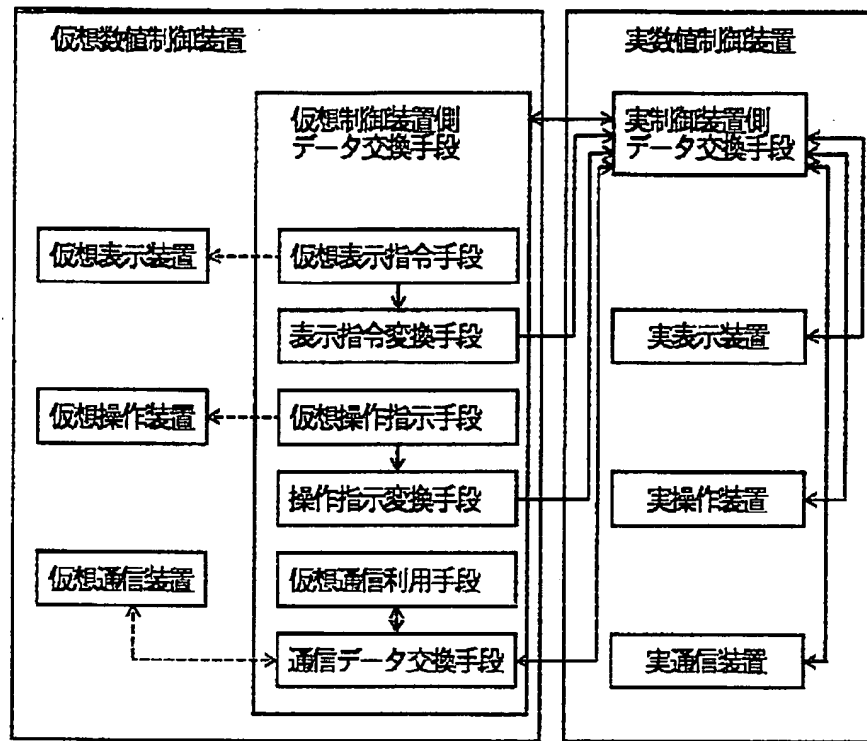
【図8】



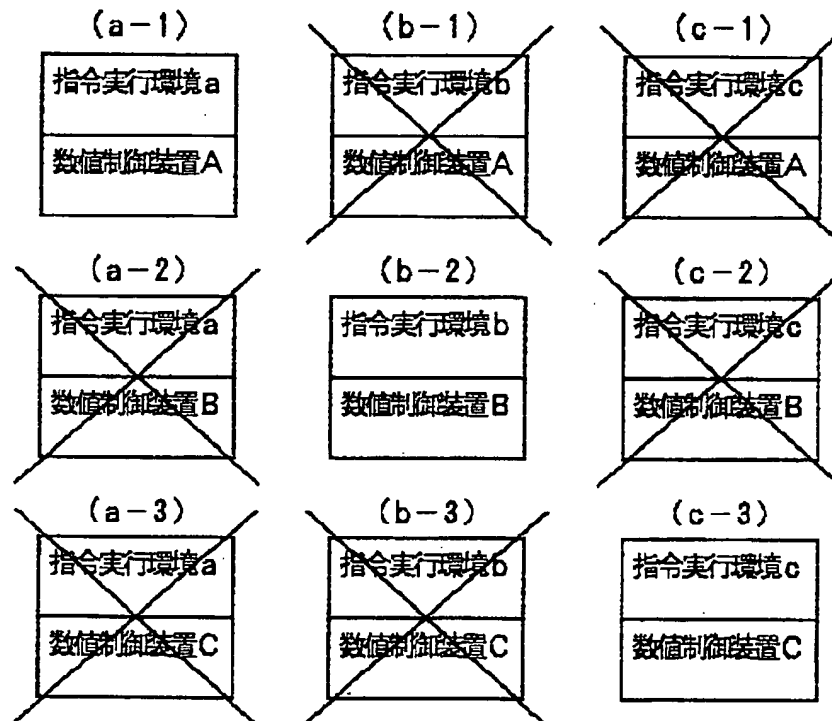
【図3】



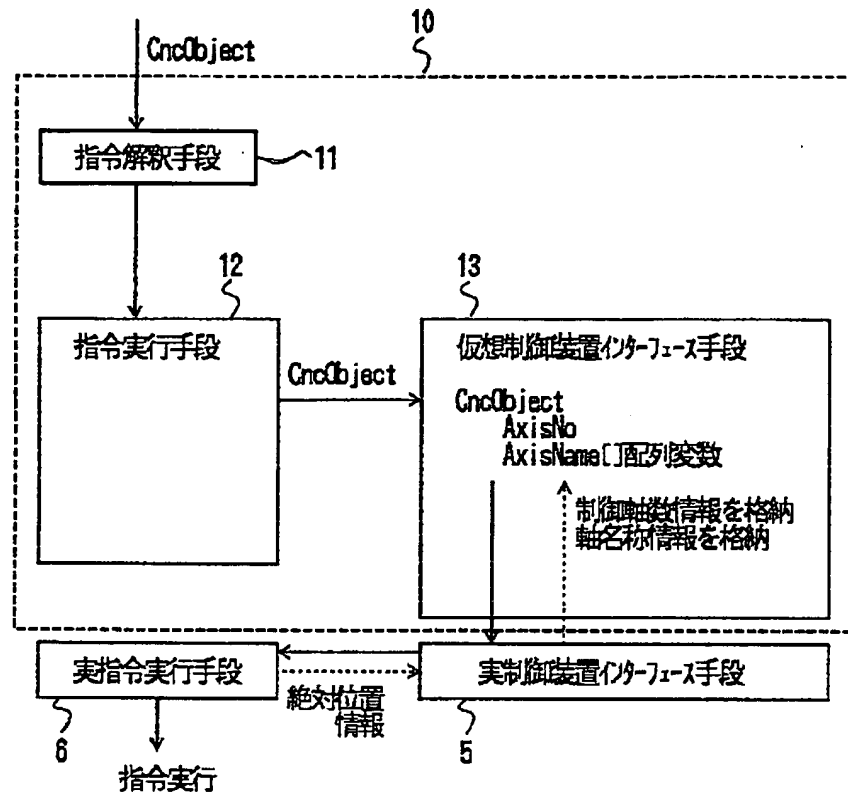
【図4】



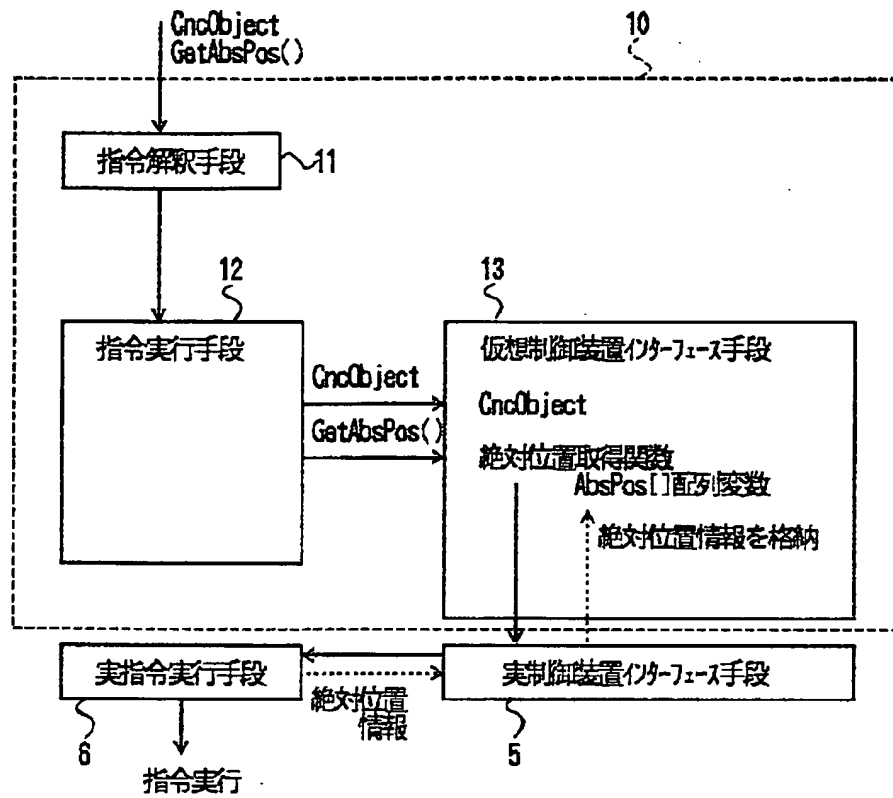
【図10】



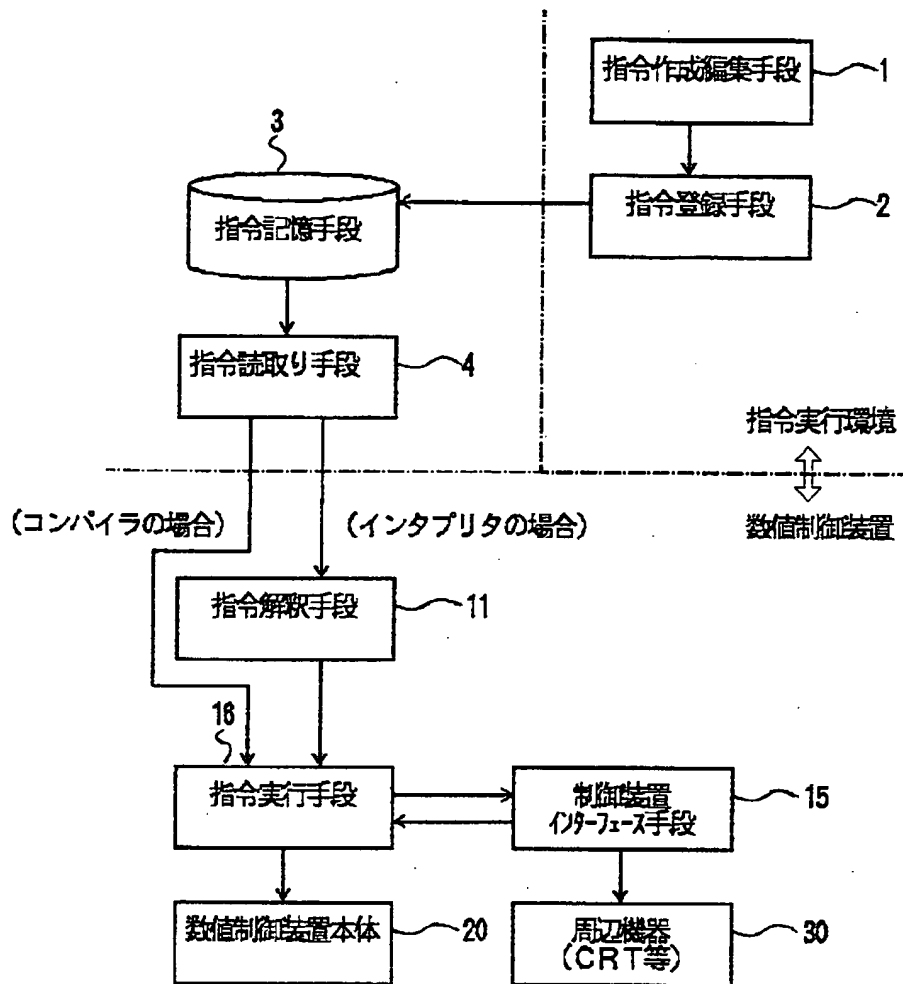
【図6】



【図7】



【図9】



【手続補正書】

【提出日】平成10年10月20日

【手続補正1】

【補正対象書類名】明細書

【補正対象項目名】特許請求の範囲

【補正方法】変更

【補正内容】

【特許請求の範囲】

【請求項1】 仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、入力される指令を固有の指令に変換する変換手段と、該変換された指令を実行する指令実行手段と、実数値制御装置とのインターフェース手段から構成され、該指令実行手段の実行結果に基づいて、該インターフェー

ス手段を介して実数値制御装置を動作させる、数値制御装置。

【請求項2】 仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、入力される指令と独立した固有の指令形態を有し、入力した指令を固有の形態の指令に変換し、該変換した指令に基づいて実数値制御装置の実行を行う、数値制御装置。

【請求項3】 仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、実数値制御装置から独立した固有の指令形態を有し、該仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、該指令の実行に基づいて実数

値制御装置の実行を行う、数値制御装置。

【請求項4】 仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、入力される指令と独立し、又実数値制御装置から独立した固有の指令形態を有し、入力した指令を固有の形態の指令に変換し、仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、該指令の実行に基づいて実数値制御装置の実行を行う、数値制御装置。

【請求項5】 前記仮想数値制御装置は、任意の言語形式及び任意の動作仕様で作成された指令を解釈し、数値制御装置固有の形態の指令に変換する指令解釈手段を備える請求項1、2または4記載の数値制御装置。

【請求項6】 前記仮想数値制御装置は、仮想数値制御装置に固有の形態に変換した指令を逐次読み込み実行する指令実行手段を備える請求項2、4、5又は6記載の数値制御装置。

【請求項7】 前記仮想数値制御装置は、仮想数値制御装置側のデータ仕様を実数値制御装置のデータ仕様に変換し、仮想数値制御装置と実数値制御装置との間のデータの整合性を調整する仮想数値制御装置インターフェース手段を備える請求項3、又は4記載の数値制御装置。

【請求項8】 前記任意の言語形式は、汎用プログラミング言語である請求項5記載の数値制御装置。

【請求項9】 表示指令の解釈と実行機能と実表示装置とのインターフェース機能を有する仮想表示装置に対して、表示指令を出力する仮想表示指令手段及び前記表示指令を実表示装置に対する表示指令に変換する表示指令変換手段と、操作指示の解釈と実行機能と実操作装置とのインターフェース機能を有する仮想操作装置に対して、操作指示を出力する仮想操作指示手段及び実操作装置から操作指示を取得し仮想操作装置に対する操作指示に変換する操作指示変換手段と、通信指令の解釈と実行機能と実通信装置とのインターフェース機能を有する仮想通信装置を有する仮想通信装置と実通信装置との間で通信データ交換を行う通信データ交換手段と、仮想数値制御装置と実数値制御装置との間でデータ交換を行う、仮想数値制御装置側のデータ交換手段及び実数値制御装置側のデータ交換手段の少なくとも1つを備えた数値制御装置。

【請求項10】 入力された指令を該指令と独立した仮想数値制御装置に固有の指令形態に変換し、該変換した指令を仮想数値制御装置上で実行することにより、実数値制御装置の実行を行う数値制御装置の実行方法。

【請求項11】 実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置上において仮想数値制御装置に固有の形態の指令を実行し、該仮想数値制御装置上の指令実行に基づいて実数値制御装置の実行を行う数値制御装置の実行方法。

【手続補正2】

【補正対象書類名】明細書

【補正対象項目名】0016

【補正方法】変更

【補正内容】

【0016】そこで、本発明の数値制御装置は、仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、入力される指令を固有の指令に変換する変換手段と、核変換された指令を実行する指令実行手段と、実数値制御装置とのインターフェース手段から構成され、該指令実行手段の実行結果に基づいて、該インターフェース手段を介して実数値制御装置を動作させる。これによって、指令や数値制御装置の仕様の相違を固有の指令によって共通化し、数値制御装置のハードウェアやソフトウェアに依存せずに数値制御装置の実行を行うことができる（請求項1に対応）。又、本発明の数値制御装置は、仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、入力される指令と独立した固有の指令形態を有し、入力した指令を固有の形態の指令に変換し、変換した指令に基づいて実数値制御装置の実行を行う。これによって、仮想数値制御装置を指令を含むアプリケーションと独立したものとし、入力した指令を仮想数値制御装置に固有の形態の指令に変換し、この変換した指令に基づいて実数値制御装置の実行を行う。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、数値制御装置の実行を行うことができる（請求項2に対応）。

【手続補正3】

【補正対象書類名】明細書

【補正対象項目名】0017

【補正方法】変更

【補正内容】

【0017】又、本発明の数値制御装置は、仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、実数値制御装置から独立した固有の指令形態を有し、この仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、この指令の実行に基づいて実数値制御装置とデータ交換を行い、実数値制御装置の実行を行う数値制御装置である。これによって、数値制御装置のハードウェアやソフトウェアに依存することなく指令に実行を行うことができる（請求項3に対応）。

【手続補正4】

【補正対象書類名】明細書

【補正対象項目名】0018

【補正方法】変更

【補正内容】

【0018】又、本発明の数値制御装置は、仮想数値制御装置と実数値制御装置から構成される数値制御装置であって、仮想数値制御装置は、入力される指令と独立し、又実数値制御装置から独立した固有の指令形態を有

し、入力した指令を固有の形態の指令に変換し、仮想数値制御装置上において仮想数値制御装置固有の形態の指令を実行し、この指令の実行に基づいて実数値制御装置とデータ交換を行い、実数値制御装置の実行を行う数値制御装置である。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、数値制御装置の実行を行うことができるとともに、数値制御装置のハードウェアやソフトウェアに依存することなく指令を実行を行うことができる（請求項4に対応）。

【手続補正5】

【補正対象書類名】明細書

【補正対象項目名】0020

【補正方法】変更

【補正内容】

【0020】なお、指令を解釈する機能及び数値制御装置固有の形態の指令へ変換する機能は、ソフトウェアにより行うことができ、又、両指令間の対応関係データは、ROM等の記憶装置に格納することができる（請求項5に対応）。

【手続補正6】

【補正対象書類名】明細書

【補正対象項目名】0021

【補正方法】変更

【補正内容】

【0021】又、本発明の数値制御装置において、仮想数値制御装置は、仮想数値制御装置に固有の形態に変換した指令を逐次読み込み実行する指令実行手段を備える構成とする。指令実行手段は、前記指令解釈手段で解釈し変換した指令を、通常のプロセッサの動作と同様に逐次読み出して実行する。このとき、指令実行手段は仮想数値制御装置に固有の指令を実行するため、指令の仕様や指令を記述するプログラミング言語に依存せずに行うことができる（請求項6に対応）。

【手続補正7】

【補正対象書類名】明細書

【補正対象項目名】0022

【補正方法】変更

【補正内容】

【0022】又、本発明の数値制御装置において、仮想数値制御装置は、仮想数値制御装置側のデータ仕様を実数値制御装置のデータ仕様に変換し、仮想数値制御装置と実数値制御装置との間のデータの整合性を調整する仮想数値制御装置インターフェース手段を備える構成とする。仮想数値制御装置インターフェース手段は、前記指令実行手段の指令の形態を実数値制御装置が実行可能な形態に変換し、変換した指令を実数値制御装置に渡す。仮想数値制御装置インターフェース手段は、仮想数値制御装置に固有の指令形態と実数値制御装置側の指令形態との対応関係を有しており、両者間の整合性をとる。これによって、数値制御装置のハードウェアやソフトウェ

アに依存することなく指令に実行を行うことができる（請求項7に対応）。

【手続補正8】

【補正対象書類名】明細書

【補正対象項目名】0023

【補正方法】変更

【補正内容】

【0023】又、本発明の数値制御装置において、入力される指令を表すプログラミング言語形式は、汎用プログラミング言語とするものである。汎用プログラミング言語の例としては、例えばC++言語やJAVA言語を用いることができる。この他、入力される指令をGコード等のコード命令とすることもできる（請求項8に対応）。

【手続補正9】

【補正対象書類名】明細書

【補正対象項目名】0024

【補正方法】変更

【補正内容】

【0024】さらに、本発明の数値制御装置において、表示指令の解釈と実行機能と実表示装置とのインターフェース機能を有する仮想表示装置に対して、表示指令を出力する仮想表示指令手段及び前記表示指令を実表示装置に対する表示指令に変換する表示指令変換手段と、操作指令の解釈と実行機能と実操作装置とのインターフェース機能を有する仮想操作装置に対して、操作指示を出力する仮想操作指示手段及び実操作装置から操作指示を取得し仮想操作装置に対する操作指示に変換する操作指示変換手段と、通信指令の解釈と実行機能と実通信装置とのインターフェース機能を有する仮想通信装置と実通信装置との間で通信データ交換を行う通信データ交換手段と、仮想数値制御装置と実数値制御装置との間でデータ交換を行う、仮想数値制御装置側のデータ交換手段及び実数値制御装置側のデータ交換手段の少なくとも1つを備えた構成とすることができる。

【手続補正10】

【補正対象書類名】明細書

【補正対象項目名】0025

【補正方法】変更

【補正内容】

【0025】この構成によれば、数値制御装置の表示機能や操作機能や通信機能や、その他数値制御装置に係わるデータ交換の機能を、仮想数値制御装置上で実行して実数値制御装置側に反映させることができる（請求項9に対応）。

【手続補正11】

【補正対象書類名】明細書

【補正対象項目名】0027

【補正方法】変更

【補正内容】

【0027】又、本発明は、入力された指令を該指令と独立した仮想数値制御装置に固有の指令形態に変換し、該変換した指令を仮想数値制御装置上で実行することにより、実数値制御装置の実行を行う数値制御装置の実行方法である。この方法によって、仮想数値制御装置を指令を含むアプリケーションと独立したものとし、入力した指令を仮想数値制御装置に固有の形態の指令に変換し、この変換した指令に基づいて実数値制御装置の実行することができる。これによって、指令の仕様や指令を記述するプログラミング言語に依存せずに、数値制御装置の実行を行うことができる（請求項10に対応）。

【手続補正12】

【補正対象書類名】明細書

【補正対象項目名】0028

【補正方法】変更

【補正内容】

【0028】又、本発明は、実数値制御装置から独立した固有の指令形態を有する仮想数値制御装置上において仮想数値制御装置に固有の形態の指令を実行し、該仮想数値制御装置上の指令実行に基づいて実数値制御装置の実行を行う数値制御装置の実行方法である。この方法によって、数値制御装置のハードウェアやソフトウェアに依存することなく指令に実行を行うことができる（請求項11に対応）。